Image Classification by a Two Dimensional Hidden Markov Model

Jia Li, Amir Najmi and Robert M. Gray *

November 25, 1998

Abstract

Conventional block-based image classification algorithms, such as CART and VQ based classification, ignore the statistical dependency among image blocks. Consequently, these algorithms often suffer from over-localization. In order to benefit from the inter-block dependency, an image classification algorithm based on a hidden Markov model (HMM) is developed. An HMM for image classification, a two dimensional extension of the one dimensional HMM used for speech recognition, has transition probabilities conditioned on the states of neighboring blocks from both directions. Thus, the dependency in two dimensions can be reflected simultaneously. The HMM parameters are estimated by the EM algorithm. A two dimensional version of the Viterbi algorithm is also developed to classify optimally an image based on the trained HMM. An application of the HMM algorithm to document image and aerial image segmentation shows that the algorithm outperforms CART and Bayes VQ.

I Introduction

For most block based image classification algorithms, such as CART [1], images are divided into blocks and decisions are made independently for the class of each block. This approach leads to an issue of choosing block sizes. We do not want to choose a too large block size since this obviously causes crude classification. On the other hand, if we choose a small block size, only very local properties belonging to the small block are examined in classification. The penalty then comes from losing information about surrounding regions. A well known method in signal processing to attack this type of problem is to use context information. Trellis coding [2] in image compression is such an example. How to introduce "context" into classifiers is what is of interest to us. Previous work [3, 4] has looked into ways of taking advantage of context information to improve classification performance for document image segmentation. Both block sizes and classification rules can vary according to context. The improvement achieved demonstrates the potential of context information to help classification. The purpose of this paper is to introduce a two dimensional hidden Markov model (2-D HMM) as a general framework for context dependent classifiers.

The theory of hidden Markov models in one dimension (1-D HMMs) was developed in the 1960s by Baum, Eagon, Petrie, Soules, and Weiss [5, 6, 7, 8]. HMMs have earned their popularity mostly from successful application to speech recognition [9, 10, 11, 12, 13]. Underlying an HMM is a basic Markov chain [14]. In fact, an HMM is simply a "Markov Source" as defined by Gallager [15]: a conditionally independent process on a Markov chain or, equivalently, a Markov chain viewed

*The authors are with the Information Systems Laboratory, Department of Electrical engineering, Stanford University, CA 94305, U.S.A. Email: jiali@isl.stanford.edu, zaalim@leland.stanford.edu, rmgray@stanford.edu. This work was supported by the National Science Foundation under NSF Grant No. MIP-931190 and by gifts from Hewlett-Packard, Inc., and SK Telecom, Inc.

through a memoryless channel. Thus, at any discrete unit of time the system is assumed to exist in one of a finite set of states. Transitions between states take place according to a fixed probability depending only on the state of the system at the unit of time immediately preceding (1-step Markovian). In an HMM, at each unit of time a single observation is generated from the current state according to a probability distribution depending only on the state. Thus in contrast to a Markov model, since the observation is a random function of the state, it is not in general possible to determine the current state by simply looking at the current observation. HMMs owe both their name and modeling power to the fact that these states represent abstract quantities that are themselves never observed. They correspond to "clusters" of contexts having similar probability distributions of the observation.

Suppose that there are M states $\{1, ..., M\}$ and that the probability of transition between states i and j is a_{ij}. Hence the probability that at time t the system will be in the state j given that at time $t-1$ it was in state i is a_{ij}. Define x_t as the observation of the system at time t. This observation is generated according to a probability distribution dependent only on the state at time t. Let $b_i(x)$ be the probability distribution of x in state i. If π_i is the probability of being in state i at time $t = 1$, then the likelihood of observing the sequence $\{x_t\}_{t=1}^T$ is given by

$$L(x) = \sum_{s_1, s_2, .., s_T} \pi_{s_1} b_{s_1}(x_1) a_{s_1 s_2} b_{s_2}(x_2) .. a_{s_{T-1} s_T} b_T(x_T)$$

where s_t represents the state at time t. The above formula can be evaluated more efficiently by the following recursive formula, the so-called *forward procedure*:

$$L(x) = \sum_{i=1}^{M} \theta_T(i)$$

$$\theta_1(i) = \pi_i b_i(x_1) \qquad 1 \le i \le M$$

$$\theta_{t+1}(j) = b_j(x_{t+1}) \sum_{i=1}^{M} \theta_t(i) a_{ij} \qquad 1 \le t \le T-1, 1 \le j \le M$$

$\theta_t(i)$ is the likelihood of the observations $x_1 .. x_t$ given that at time t the state is i.

For details, see any of the references on speech recognition [10, 11, 12, 16].

Of particular interest to this paper is the most likely sequence of states $\{s_t^*\}_{t=1}^T$ given the observation sequence $\{x_t\}_{t=1}^T$. This is typically computed using dynamic programming (the Viterbi algorithm [17]).

Estimation of 1-D HMM model parameters is usually performed according to the Baum-Welch algorithm (a special case of the EM algorithm [18]). Details of the iterative algorithm can be found in any of the references on speech recognition. One way to think of this algorithm is as a reestimation of the state parameters from each training observation sequence, weighted by the likelihood of each possible state sequence that could have caused it. This results in maximum likelihood parameter estimates. An approximation to maximum likelihood training is what is often termed Viterbi training [16], in which each observation is assumed (with weight of 1) to have resulted from the single most likely state sequence that might have caused it. While more efficient computationally, Viterbi training does not in general result in maximum likelihood estimates. Note that an intermediate technique often used is to consider only the N most likely state sequences for each observation sequence for likelihood weighted training.

To apply the HMM to images, previous work extended the 1-D HMM to the pseudo 2-D HMM [19, 20]. The model is pseudo 2-D in the sense that it is not a fully connected 2-D HMM.

The basic assumption is that there exists a set of "superstates" that are Markovian. Within each superstate there is a set of simple Markovian states. For 2-D images, first the superstate is chosen using a first order Markov transition probability based on the previous superstate. This superstate determines the simple Markov chain to be used by the entire row. A simple Markov chain is then used to generate observations in the row. Thus, superstates relate to rows and simple states to columns. In particular applications, this model works better than the 1-D HMM [19], but we expect the pseudo 2-D HMM to be much more effective with regular images, such as documents. Since the effect of the state of a pixel on the state below it is distributed across the whole row, the pseudo 2-D model is too constrained for normal image classification. We thus propose using a truly 2-D HMM.

The 2-D HMM for images assumes that the probability of the system entering a particular state depends upon the state of the system at the adjacent observations in both horizontal and vertical directions. A transition from any state to any state is allowed. As in the case of speech, the probability distribution of the feature vector is modeled as a fixed Gaussian distribution for any given state. The extension is non-trivial since certain entities that correspond to single states in the 1-D case may correspond to entire state sequences in 2-D. The main difficulty with using a genuine 2-D HMM is its computational complexity, so we develop approximation methods for estimating and applying the 2-D HMM in order to achieve computational feasibility. In our applications, simulations show that classification performance approaches limits rather quickly with increasing approximation accuracy. Hence, the benefit of using the 2-D HMM is realized at reasonably low computational complexity.

In Section II, we provide a mathematical formulation of the basic assumptions of the 2-D HMM. Section III derives the iterative estimation algorithm for the model according to the general EM algorithm. Computational complexity is analyzed in Section IV. In Section IV, backward and forward probabilities in the 2-D case are introduced to efficiently estimate the model. Our algorithm further lowers the computational complexity by using the Viterbi training. The 2-D version of the Viterbi algorithm is described in Section V. Two applications of classification based on the 2-D HMM are presented in Section VI. We conclude in Section VII.

II Basic Assumptions of 2-D HMM

As in all block based classification systems, an image to be classified is divided into blocks and feature vectors are evaluated as the statistics of the blocks. The image is then classified according to the feature vectors.

The 2-D HMM assumes that the feature vectors are generated by a Markov model which may change state once every block. Suppose there are M states, $\{1, ..., M\}$, the state of block (i, j) is denoted by $s_{i,j}$. The feature vector of block (i, j) is $\mathbf{v}_{i,j}$ and the class is $c_{i,j}$. We use $P(\cdot)$ to represent the probability or likelihood of an event. We denote $(i', j') < (i, j)$, or $(i, j) > (i', j')$, if $i' < i$ or $i' = i$, and $j' < j$; in which case we say that block (i', j') is before block (i, j). For example, in the left panel of Fig. 1, the blocks before (i, j) are the shaded blocks. This sense of order is the same as the raster order of row by row. We would like to point out, however, that we introduce this order only for stating the assumptions. In classification, we do not classify blocks one by one in such an order. Our classification algorithm tries to find the optimal combination of classes jointly for many blocks at once. A one dimensional approach of joint classification, assuming a scanning order in classification, is usually suboptimal.

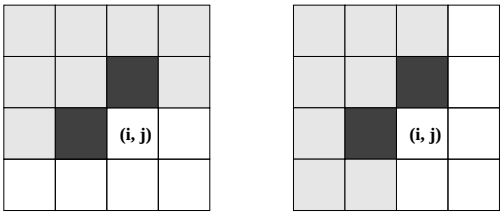

Figure 1: Markovian property of the transition of states

The first assumption we make is that

$$\begin{aligned} P(s_{i,j}|\Psi) &= a_{m,n,l} \quad , \\ \text{where} \quad \Psi &= \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j') < (i,j)\} \\ \text{and} \quad m = s_{i-1,j},\ n &= s_{i,j-1}, \text{ and } l = s_{i,j} \quad . \end{aligned}$$

The above assumption can be summarized by two points. First, the state $s_{i',j'}$ is a sufficient statistic for $(s_{i',j'}, \mathbf{v}_{i',j'})$ for estimating transition probabilities, i.e., $\mathbf{v}$ are conditionally memoryless. Second, the state transition is first order Markovian in a two dimensional sense. Shown in the left panel of Fig. 1, knowing the states of all the shaded blocks, we only need the states of the two adjacent blocks in the darker shade to calculate the transition probability to a next state. We also assume that there is a unique mapping from states to classes. Thus, the classes of the blocks are determined once the states are known.

The second assumption is that for every state, the feature vectors follow a Gaussian mixture distribution. Once the state of a block is known, the feature vector is conditionally independent of the other blocks. Since any state with an M-component Gaussian mixture can be split into M substates with single Gaussian distributions, we restrict ourselves to single Gaussian distributions. For a block with state s and feature vector $\mathbf{x}$, the distribution is

$$b_s(\mathbf{v}) = \frac{1}{\sqrt{(2\pi)^n |\mathbf{\Sigma}_s|}} e^{-\frac{1}{2}(\mathbf{v}-\mu_s)' \mathbf{\Sigma}_s^{-1} (\mathbf{v}-\mu_s)} \quad ,$$

where $\mathbf{\Sigma}_s$ is the covariance matrix and μ_s is the mean vector.

The Markovian assumption on state transitions can simplify significantly the evaluation of the probability of the states, i.e., $P\{s_{i,j}, (i,j) \in \mathbb{N}\}$, where $\mathbb{N} = \{(i,j), 0 \le i < \mathfrak{m}, 0 \le j < z\}$ refers to all the blocks in an image. To expand efficiently this probability by the conditional probability formula, we first prove that a rotated form of the two dimensional Markovian property holds given the two assumptions. Recall that we define $(i',j') < (i,j)$ if $i' < i$ or $i' = i$, and $j' < j$. We then define a rotated relation of "$<$", denoted by "$\tilde{<}$", which specifies $(i',j')\tilde{<}(i,j)$, or $(i,j)\tilde{>}(i',j')$, if $j' < j$ or $j' = j$, and $i' < i$. An example is shown in the right panel of Fig. 1. To prove that

$$\begin{aligned} P(s_{i,j} \,|\, \tilde{\Psi}) &= a_{m,n,l} \quad , \\ \text{where} \quad \tilde{\Psi} &= \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j')\tilde{<}(i,j)\} \\ \text{and} \quad m = s_{i-1,j},\ n &= s_{i,j-1}, \text{ and } l = s_{i,j} \quad , \end{aligned}$$

we define $\Psi = \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j') < (i,j)\}$ and introduce the following notation:

$$\begin{aligned} \tilde{\Psi} \cup \Psi &= \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j') < (i,j) \text{ or } (i',j')\tilde{<}(i,j)\} \quad , \\ \tilde{\Psi} \cap \Psi &= \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j') < (i,j) \text{ and } \ (i',j')\tilde{<}(i,j)\} \quad , \end{aligned}$$

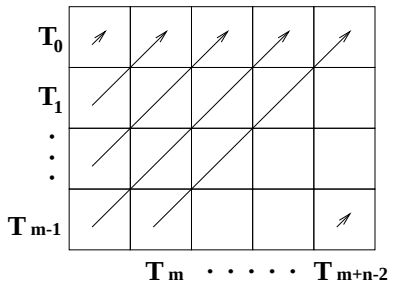

Figure 2: Blocks on diagonals of an image

$$\tilde{\Psi} - \Psi = \{s_{i',j'}, \mathbf{v}_{i',j'}, (i',j') \tilde{<} (i,j) \text{ and } (i',j') > (i,j)\} \quad .$$

Note that $\tilde{\Psi} = (\tilde{\Psi} \cup \Psi) \cup (\tilde{\Psi} - \Psi)$. We can then derive

$$\begin{aligned}
P(s_{i,j} \,|\, \tilde{\Psi}) &= \frac{P(s_{i,j}, \tilde{\Psi} \cap \Psi, \tilde{\Psi} - \Psi)}{P(\tilde{\Psi})} \\
&= \frac{P(\tilde{\Psi} \cap \Psi) P(s_{i,j} \,|\, \tilde{\Psi} \cap \Psi) P(\tilde{\Psi} - \Psi \,|\, \tilde{\Psi} \cap \Psi, s_{i,j})}{P(\tilde{\Psi})} \qquad (1) \\
&= \frac{P(\tilde{\Psi} \cap \Psi) P(s_{i,j} \,|\, s_{i-1,j}, s_{i,j-1}) P(\tilde{\Psi} - \Psi \,|\, \tilde{\Psi} \cap \Psi, s_{i,j})}{P(\tilde{\Psi})} \qquad (2) \\
&= \frac{P(s_{i,j} \,|\, s_{i-1,j}, s_{i,j-1}) P(\tilde{\Psi} \cap \Psi) P(\tilde{\Psi} - \Psi \,|\, \tilde{\Psi} \cap \Psi)}{P(\tilde{\Psi})} \qquad (3) \\
&= P(s_{i,j} \,|\, s_{i-1,j}, s_{i,j-1}) \\
&= a_{m,n,l}
\end{aligned}$$

where $m = s_{i-1,j}$, $n = s_{i,j-1}$, and $l = s_{i,j}$. Equality (1) follows from the expansion of conditional probability. Equality (2) follows from the Markovian assumption. Equality (3) holds due to both the Markovian assumption and the assumption that the feature vector of a block is conditionally independent of other blocks given its state.

According to the derivation, there actually exists a stronger statement, which is $P(s_{i,j} \,|\, \tilde{\Psi} \cup \Psi) = P(s_{i,j} \,|\, s_{i-1,j}, s_{i,j-1})$. The reason is that in the derivation, if we change $\tilde{\Psi} \cap \Psi$ to Ψ and $\tilde{\Psi}$ to $\tilde{\Psi} \cup \Psi$, all the equalities still hold. This property implies the original Markovian assumption and its rotated version. We refer to this property as a stronger Markovian property.

We now simplify the expansion of $P\{s_{i,j}, (i,j) \in \mathbb{N}\}$:

$$P\{s_{i,j}, (i,j) \in \mathbb{N}\} = P(T_0) \cdot P(T_1 \,|\, T_0) \cdots P(T_{\mathfrak{m}+z-2} \,|\, T_{\mathfrak{m}+z-3}, T_{\mathfrak{m}+z-4}, \ldots, T_0) \quad , \qquad (4)$$

where T_i denotes the sequence of states for blocks lying on diagonal i, i.e., $(s_{i,0}, s_{i-1,1}, \cdots, s_{0,i})$, as shown in Fig. 2.

We next show that $P(T_i \,|\, T_{i-1}, \ldots, T_0) = P(T_i \,|\, T_{i-1})$. Without loss of generality, suppose $T_i = \{s_{i,0}, s_{i-1,1}, \ldots, s_{0,i}\}$, then $T_{i-1} = \{s_{i-1,0}, s_{i-2,1}, \ldots, s_{0,i-1}\}$ and

$$\begin{aligned}
P(T_i \,|\, T_{i-1}, \ldots, T_0) &= P(s_{i,0}, s_{i-1,1}, \ldots, s_{0,i} \,|\, T_{i-1}, T_{i-2}, \ldots, T_0) \\
&= P(s_{i,0} \,|\, T_{i-1}, \ldots, T_0) \cdot P(s_{i-1,1} \,|\, s_{i,0}, T_{i-1}, \ldots, T_0) \\
&\quad \cdots P(s_{0,i} \,|\, s_{1,i-1}, \ldots, s_{i,0}, T_{i-1}, \ldots, T_0) \\
&= P(s_{i,0} \,|\, s_{i-1,0}) \cdot P(s_{i-1,1} \,|\, s_{i-2,1}, s_{i-1,0}) \cdots P(s_{0,i} \,|\, s_{0,i-1}) \quad .
\end{aligned}$$

The last equality is obtained from the stronger Markovian property. Since all the states $s_{i,j}$ which appear in the conditions are in T_{i-1}, we conclude that

$$P(T_i \,|\, T_{i-1}, \ldots, T_0) \quad = \quad P(T_i \,|\, T_{i-1}) \quad .$$

Equation (4) is simplified to

$$P\{s_{i,j}, (i,j) \in \mathbb{N}\} = P(T_0) \cdot P(T_1 \,|\, T_0) \cdots P(T_{\mathrm{m}+z-2} \,|\, T_{\mathrm{m}+z-3}) \quad . \tag{5}$$

The state sequence T_i thus serves as an "isolating" element in the expansion of $P(s_{i,j}, (i,j) \in \mathbb{N})$, which plays the role of a state at a single unit of time in the case of a one dimensional Markov model. As we shall see, this property is essential for developing our algorithm. We may notice that, other than diagonals, there exist geometric forms which can serve as "isolating" elements as well, for example, state sequences on rows or columns. We prefer state sequences T_i on diagonals because conditioned on T_{i-1}, all the states in T_i are statistically independent. This fact is a direct result of the stronger Markovian property. The conditional independence of the states in T_i given T_{i-1} significantly reduces computation.

The task of our classifier is to estimate the 2-D HMM from training data and to classify images by finding the combination of states with the maximum posterior probability given the observed feature vectors.

We point out that the underlying state process we have defined is a special Markov random field (MRF) [21, 22], which was referred to as Markov Mesh and proposed by Abend, Harley and Kanal [23, 24] for the classification of binary random patterns. It is an extension of a Markov chain into two dimensions so that a much larger class of spatial dependencies can be taken into consideration. In our assumption of the Markovian property, the states in condition are the states of blocks above and to the left of a current block, thus there exists a certain sense of "past". This type of Markov random fields is called the "causal" MRF [25, 24, 26]. The causality enables the derivation of an analytic iterative algorithm to estimate an HMM and to estimate states with the maximum a posteriori probability.

III Estimation of the Model

For the assumed HMM, we need to estimate the following parameters: transition probabilities $a_{m,n,l}$, where $m, n, l = 1, ..., M$, and M is the total number of states, the mean vectors μ_m, and the covariance matrices $\mathbf{\Sigma}_m$ of the Gaussian distributions, $m = 1, ..., M$. We define the set $\mathcal{M} = \{1, ..., M\}$. The parameters are estimated by the maximum likelihood criterion (ML) using the EM algorithm [18, 27, 8]. We first introduce briefly the EM algorithm as described in Dempster, Laird and Rubin [18]. Then we apply the algorithm to our case to derive a specific formula.

The EM algorithm provides an iterative computation of maximum likelihood estimation in the case of the observed data being incomplete. The term "incomplete" reflects the fact that we need to estimate the distribution of $\mathbf{x}$, in sample space $\mathcal{X}$, but we can only observe $\mathbf{x}$ indirectly through $\mathbf{y}$, in sample space $\mathcal{Y}$. In many cases, there is a mapping $\mathbf{x} \to \mathbf{y}(\mathbf{x})$ from $\mathcal{X}$ to $\mathcal{Y}$, and $\mathbf{x}$ is only known to lie in a subset of $\mathcal{X}$, denoted by $\mathcal{X}(\mathbf{y})$, which is determined by the equation $\mathbf{y} = \mathbf{y}(\mathbf{x})$. We postulate a family of distribution $f(\mathbf{x} \,|\, \phi)$, with parameters $\phi \in \Omega$, on $\mathbf{x}$. The distribution of $\mathbf{y}$, $g(\mathbf{y} \,|\, \phi)$, can be derived as

$$g(\mathbf{y} \,|\, \phi) \quad = \quad \int_{\mathcal{X}(\mathbf{y})} f(\mathbf{x} \,|\, \phi) d\mathbf{x} \quad .$$

The EM algorithm is aimed at finding a ϕ which maximizes $g(\mathbf{y} \mid \phi)$ given an observed $\mathbf{y}$.

Before we describe the algorithm, we introduce a function [18]

$$Q(\phi' \mid \phi) \quad = \quad E(\log f(\mathbf{x} \mid \phi') \mid \mathbf{y}, \phi) \quad ,$$

which is assumed to exist for all pairs (ϕ', ϕ). In particular, we assume that $f(\mathbf{x} \mid \phi) > 0$ almost everywhere in $\phi \in \Omega$. The function $Q(\phi' \mid \phi)$ is the expected value of $\log f(\mathbf{x} \mid \phi')$ with the distribution of $\mathbf{x}$ being the conditional distribution given $\mathbf{y}$ and parameter ϕ. The EM iteration $\phi^{(p)} \rightarrow \phi^{(p+1)}$ is defined in [18] as follows

1. E-step: Compute $Q(\phi \mid \phi^{(p)})$.
2. M-step: Choose $\phi^{(p+1)}$ to be a value of $\phi \in \Omega$ which maximizes $Q(\phi \mid \phi^{(p)})$.

We define the following notation.

1. The set of observed feature vectors for the whole image is $\bar{\mathbf{v}} = \{\mathbf{v}_{i,j}, (i,j) \in \mathbb{N}\}$.
2. The states for the image are $\mathbf{s} = \{s_{i,j}, (i,j) \in \mathbb{N}\}$.
3. The classes for the image are $\mathbf{c} = \{c_{i,j}, (i,j) \in \mathbb{N}\}$.
4. The mapping from a state m to its class is $C(m)$, and the set of classes mapped from states $\mathbf{s}$ is denoted by $C(\mathbf{s})$.

Specific to our case, the complete data $\mathbf{x}$ are $\{s_{i,j}, \mathbf{v}_{i,j}, (i,j) \in \mathbb{N}\}$ and the incomplete data $\mathbf{y}$ are $\{c_{i,j}, \mathbf{v}_{i,j}, (i,j) \in \mathbb{N}\}$. The function $f(\mathbf{x} \mid \phi')$ is

$$\begin{aligned} f(\mathbf{x} \mid \phi') &= P(\mathbf{s} \mid \phi') \cdot P(\bar{\mathbf{v}} \mid \mathbf{s}, \phi') \\ &= P(\mathbf{s} \mid a'_{m,n,l}, m, n, l \in \mathcal{M}) \cdot P(\bar{\mathbf{v}} \mid \mathbf{s}, \mu'_m, \mathbf{\Sigma}'_m, m \in \mathcal{M}) \\ &= \prod_{(i,j) \in \mathbb{N}} a'_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} \cdot \prod_{(i,j) \in \mathbb{N}} P(\mathbf{v}_{i,j} \mid \mu'_{s_{i,j}}, \mathbf{\Sigma}'_{s_{i,j}}) \quad . \end{aligned}$$

We then have

$$\log f(\mathbf{x} \mid \phi') \quad = \quad \sum_{(i,j) \in \mathbb{N}} \log a'_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} + \sum_{(i,j) \in \mathbb{N}} \log P(\mathbf{v}_{i,j} \mid \mu'_{s_{i,j}}, \mathbf{\Sigma}'_{s_{i,j}}) \quad . \tag{6}$$

Given $\mathbf{y}$, $\mathbf{x}$ can only take finite number of values, corresponding to different sets of states $\mathbf{s}$ which have classes consistent with $\mathbf{y}$. The distribution of $\mathbf{x}$ is

$$\begin{aligned} P(\mathbf{x} \mid \mathbf{y}, \phi^{(p)}) &= \frac{1}{\alpha} I(C(\mathbf{s}) = \mathbf{c}) \cdot P(\mathbf{s} \mid \phi^{(p)}) \cdot P(\bar{\mathbf{v}} \mid \mathbf{s}, \phi^{(p)}) \\ &= \frac{1}{\alpha} I(C(\mathbf{s}) = \mathbf{c}) \cdot \prod_{(i,j) \in \mathbb{N}} a^{(p)}_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} \cdot \prod_{(i,j) \in \mathbb{N}} P(\mathbf{v}_{i,j} \mid \mu^{(p)}_{s_{i,j}}, \mathbf{\Sigma}^{(p)}_{s_{i,j}}) \quad , \end{aligned}$$

where α is a normalization constant, and the function $I(\cdot)$ is the indicator function which equals one if its argument is true and zero otherwise. From this point, we write $P(\mathbf{x} \mid \mathbf{y}, \phi^{(p)})$ as $P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)})$, assuming that all the $\mathbf{v}_{i,j}$ in $\mathbf{x}$ are the same as those in $\mathbf{y}$, since otherwise the conditional probability of $\mathbf{x}$ given $\mathbf{y}$ is zero.

In the M-step, we set $\phi^{(p+1)}$ to the ϕ' which maximizes

$$\begin{aligned} E(\log f(\mathbf{x} \mid \phi') \mid \mathbf{y}, \phi^{(p)}) &= \frac{1}{\alpha} \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{(i,j) \in \mathbb{N}} \log a'_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} + \\ & \frac{1}{\alpha} \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{(i,j) \in \mathbb{N}} \log P(\mathbf{v}_{i,j} \mid \mu'_{s_{i,j}}, \mathbf{\Sigma}'_{s_{i,j}}) \quad . \end{aligned} \tag{7}$$

Equation (7) follows directly from (6).

We can thus maximize the two items in (7) separately by choosing $a'_{m,n,l}$ and μ'_m, $\mathbf{\Sigma}'_m$ respectively. Consider the first item

$$\begin{aligned} & \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{(i,j) \in \mathbb{N}} \log a'_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} \\ = & \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{m,n,l \in \mathcal{M}} \sum_{(i,j) \in \mathbb{N}} \log a'_{m,n,l} \cdot I(m = s_{i-1,j}, n = s_{i,j-1}, l = s_{i,j}) \\ = & \sum_{m,n,l \in \mathcal{M}} \log a'_{m,n,l} \cdot \sum_{(i,j) \in \mathbb{N}} \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot I(m = s_{i-1,j}, n = s_{i,j-1}, l = s_{i,j}) \quad . \end{aligned} \tag{8}$$

Denote $H^{(p)}_{m,n,l}(i,j) = \sum_{\mathbf{s}} I(m = s_{i-1,j}, n = s_{i,j-1}, l = s_{i,j}) P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)})$, which is the probability of being in state m at block $(i-1,j)$, state n at block $(i,j-1)$, and state l at block (i,j) given the observed feature vectors, classes, and model $\phi^{(p)}$. Expression (8) becomes

$$\sum_{m,n,l \in \mathcal{M}} \log a'_{m,n,l} \sum_{(i,j) \in \mathbb{N}} H^{(p)}_{m,n,l}(i,j) \quad .$$

To maximize it under the constraint

$$\sum_{l=1}^{M} a'_{m,n,l} = 1 \text{, for all } m, n \in \mathcal{M} \quad ,$$

use the Lagrangian multiplier and take derivatives with respect to $a'_{m,n,l}$. We then conclude that

$$a'_{m,n,l} \propto \sum_{(i,j) \in \mathbb{N}} H^{(p)}_{m,n,l}(i,j) \quad ,$$

which in turn yields

$$a'_{m,n,l} = \frac{\sum_{(i,j) \in \mathbb{N}} H^{(p)}_{m,n,l}(i,j)}{\sum_{l=1}^{M} \sum_{(i,j) \in \mathbb{N}} H^{(p)}_{m,n,l}(i,j)} \quad .$$

Now we discuss the maximization of the second term in equation (7).

$$\begin{aligned} & \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{(i,j) \in \mathbb{N}} \log P(\mathbf{v}_{i,j} \mid \mu'_{s_{i,j}}, \mathbf{\Sigma}'_{s_{i,j}}) \\ = & \sum_{\mathbf{s}} P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \sum_{m=1}^{M} \sum_{(i,j) \in \mathbb{N}} \log P(\mathbf{v}_{i,j} \mid \mu'_m, \mathbf{\Sigma}'_m) I(m = s_{i,j}) \\ = & \sum_{m=1}^{M} \sum_{(i,j) \in \mathbb{N}} \sum_{\mathbf{s}} I(m = s_{i,j}) P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)}) \cdot \log P(\mathbf{v}_{i,j} \mid \mu'_m, \mathbf{\Sigma}'_m) \quad . \end{aligned}$$

Denote $L_m^{(p)}(i,j) = \sum_{\mathbf{s}} I(m = s_{i,j}) P(\mathbf{s} \mid \mathbf{y}, \phi^{(p)})$, which is the probability of being in state m at block (i,j) given the observed feature vectors, classes and model $\phi^{(p)}$. The above expression is simplified to

$$\sum_{m=1}^{M} \sum_{(i,j)\in\mathbb{N}} L_m^{(p)}(i,j) \log P(\mathbf{v}_{i,j} \mid \mu'_m, \mathbf{\Sigma}'_m) \quad .$$

It is known that for Gaussian distributions, the ML estimation of μ'_m is the sample average of the data , and the ML estimation of $\mathbf{\Sigma}'_m$ is the sample covariance matrix of the data [28]. Since in our case, the data is weighted by $L_m^{(p)}(i,j)$, the M.L.E. of μ'_m and $\mathbf{\Sigma}'_m$ are

$$\begin{aligned} \mu'_m &= \frac{\sum_{i,j} L_m^{(p)}(i,j)\mathbf{v}_{i,j}}{\sum_{i,j} L_m^{(p)}(i,j)} \quad , \\ \mathbf{\Sigma}'_m &= \frac{\sum_{i,j} L_m^{(p)}(i,j)(\mathbf{v}_{i,j} - \mu'_m)(\mathbf{v}_{i,j} - \mu'_m)'}{\sum_{i,j} L_m^{(p)}(i,j)} \quad . \end{aligned}$$

In summary, the estimation algorithm iteratively improves the model estimation by the following two steps.

1. Given the current model estimation $\phi^{(p)}$ and the observed feature vectors $\mathbf{v}_{i,j}$ and classes $c_{i,j}$, the mean vectors and covariance matrices are updated by

$$\mu_m^{(p+1)} = \frac{\Sigma_{i,j} L_m^{(p)}(i,j)\mathbf{v}_{i,j}}{\Sigma_{i,j} L_m^{(p)}(i,j)} \tag{9}$$

$$\mathbf{\Sigma}_m^{(p+1)} = \frac{\Sigma_{i,j} L_m^{(p)}(i,j)(\mathbf{v}_{i,j} - \mu_m^{(p+1)})(\mathbf{v}_{i,j} - \mu_m^{(p+1)})'}{\Sigma_{i,j} L_m^{(p)}(i,j)} \quad . \tag{10}$$

   The probability $L_m^{(p)}(i,j)$ can be calculated by

$$\begin{aligned} L_m^{(p)}(i,j) = &\sum_{\mathbf{s}} I(m = s_{i,j}) \cdot \\ &\frac{1}{\alpha} I(C(\mathbf{s}) = \mathbf{c}) \cdot \prod_{(i,j)\in\mathbb{N}} a^{(p)}_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} \cdot \prod_{(i,j)\in\mathbb{N}} P(\mathbf{v}_{i,j} \mid \mu^{(p)}_{s_{i,j}}, \mathbf{\Sigma}^{(p)}_{s_{i,j}}) \quad . \end{aligned} \tag{11}$$

2. The transition probabilities are updated by

$$a^{(p+1)}_{m,n,l} = \frac{\sum_{i,j} H^{(p)}_{m,n,l}(i,j)}{\sum_{l=1}^{M} \sum_{i,j} H^{(p)}_{m,n,l}(i,j)} \quad ,$$

   where $H^{(p)}_{m,n,l}(i,j)$ can be calculated by

$$\begin{aligned} H^{(p)}_{m,n,l}(i,j) = &\sum_{\mathbf{s}} I(m = s_{i-1,j}, n = s_{i,j-1}, l = s_{i,j}) \cdot \\ &\frac{1}{\alpha} I(C(\mathbf{s}) = \mathbf{c}) \cdot \prod_{(i,j)\in\mathbb{N}} a^{(p)}_{s_{i-1,j}, s_{i,j-1}, s_{i,j}} \cdot \prod_{(i,j)\in\mathbb{N}} P(\mathbf{v}_{i,j} \mid \mu^{(p)}_{s_{i,j}}, \mathbf{\Sigma}^{(p)}_{s_{i,j}}) \quad . \end{aligned} \tag{12}$$

In the case of one dimensional HMM as used in speech recognition, computationally efficient formulas exist for calculating $L_m(k)$ and $H_{m,l}(k)$ [16]. For 2-D HMM, however, the computation of $L_m(i,j)$ and $H_{m,n,l}(i,j)$ is not feasible, due to the two dimensional transition probabilities. The next section will discuss why this is so and how to reduce the computational complexity.

IV Computational Complexity

As shown in previous section, the calculation of the probabilities $L_m^{(p)}(i,j)$ and $H_{m,n,l}^{(p)}(i,j)$ is the key for the iterative estimation of the model parameters. If we calculate $L_m^{(p)}(i,j)$ and $H_{m,n,l}^{(p)}(i,j)$ directly according to equation (11) and (12), we need to consider all the combinations of states which yield the same classes as those appeared in the training set. The large number of such combinations of states results in infeasible computation. Let us take $L_m^{(p)}(i,j)$ as an example. Suppose there are M_0 states for each class and the number of blocks in an image is $w \times z$ as previously assumed, then the number of admissible combinations of states which satisfy $C(\mathbf{s}) = \mathbf{c}$ and $s_{i,j} = m$, is $(w \times z - 1)^{M_0}$. When applying the HMM algorithm, although we often break one image into many sub-images such that w, or z, is the number of blocks in one column, or one row, in a sub-image, we need to keep w and z sufficiently large to ensure an adequate amount of context information being incorporated in classification. In the limit, if $w = z = 1$, the algorithm is simply a parametric classification algorithm performed independently on each block. It is normal to have $w = z = 8$. In this case, if we have 4 states for each class, the number of the combinations of states is $(w \times z - 1)^{M_0} = 63^4$, which is prohibitive for a straight-forward calculation of $L_m^{(p)}(i,j)$. Similar difficulty occurs for estimating one dimensional HMM. The problem is solved by a recursive calculation of forward and backward probabilities [16].

The idea of using forward and backward probabilities can be applied to the two dimensional HMM to simplify the computation. Recall equation (5) in Section II,

$$P(s_{i,j}, (i,j) \in \mathbb{N}) = P(T_0) \cdot P(T_1 \,|\, T_0) \cdots P(T_{w+z-2} \,|\, T_{w+z-3}) \quad .$$

The fact that the state sequence T_i on a diagonal is an "isolating" element in the expansion of $P(s_{i,j}, (i,j) \in \mathbb{N})$ enables us to define the forward and backward probabilities and to evaluate them by recursive formulas.

Let us clarify notation first. In addition to the notation we provide in the list in section III, we need to use the following definitions.

1. The diagonal on which block (i,j) lies is denoted by $\Delta(i,j)$.
2. The feature vectors on diagonal d, $\{\mathbf{v}_{i,j},\ (i,j) : \Delta(i,j) = d\}$, is denoted by $\bar{\mathbf{v}}(d)$.
3. The state sequence on diagonal d, $\{s_{i,j},\ (i,j) : \Delta(i,j) = d\}$, is denoted by $\mathbf{s}(d)$.
4. For a state sequence T on diagonal d, its value at block (i,j) is $T(i,j)$.

The forward probability $\theta_T(d)$ for some model $\mathbf{M}$ is defined as

$$\theta_T(d) = P(\mathbf{v}_{i,j}, (i,j) : \Delta(i,j) \le d,\ \mathbf{s}(d) = T \,|\, \mathbf{M})$$

The forward probability $\theta_T(d)$ is the joint distribution of observing the vectors lying on or above diagonal d and having state sequence T for blocks on diagonal d.

The backward probability $\beta_T(d)$ is defined as

$$\beta_T(d) = P(\mathbf{v}_{i,j}, (i,j) : \Delta(i,j) > d \,|\, \mathbf{s}(d) = T, \mathbf{M})$$

That is, $\beta_T(d)$ is the conditional distribution of observing the vectors lying below diagonal d given the state sequence on diagonal d is T.

Similar to the case of 1-D HMM, we can derive recursive formulas for calculating $\theta_T(d)$ and $\beta_T(d)$, which are listed below

$$\theta_{T_d}(d) = \sum_{T_{d-1}} \theta_{T_{d-1}}(d-1) \cdot P(T_d \,|\, T_{d-1}, \mathbf{M}) \cdot P(\bar{\mathbf{v}}(d) \,|\, T_d, \mathbf{M}) \quad , \tag{13}$$

$$\beta_{T_d}(d) = \sum_{T_{d+1}} P(T_{d+1} \,|\, T_d, \mathbf{M}) \cdot P(\bar{\mathbf{v}}(d+1) \,|\, T_{d+1}, \mathbf{M}) \cdot \beta_{T_{d+1}}(d+1) \quad . \tag{14}$$

We can then calculate $L_m(i,j)$ given model $\mathbf{M}$ by

$$\begin{aligned} L_m(i,j) &= P(s_{i,j} = m \,|\, \bar{\mathbf{v}}, \mathbf{c}, \mathbf{M}) \\ &= \begin{cases} \sum_{T_d : T_d(i,j)=m} P(T_d \,|\, \bar{\mathbf{v}}, \mathbf{c}, \mathbf{M}) & , \text{ if } C(m) = c_{i,j} \\ 0 & , \text{ otherwise} \end{cases} . \end{aligned}$$

Let us consider the case $C(m) = c_{i,j}$. It is assumed in the derivation below that the summation over T_d only covers T_d which yields consistent classes with the training data.

$$\begin{aligned} L_m(i,j) &= \sum_{T_d : T_d(i,j)=m} \frac{P(T_d, \bar{\mathbf{v}} \,|\, \mathbf{M})}{P(\bar{\mathbf{v}}, \mathbf{c} \,|\, \mathbf{M})} \\ &= \sum_{T_d : T_d(i,j)=m} \frac{\theta_{T_d}(\Delta(i,j)) \cdot \beta_{T_d}(\Delta(i,j))}{P(\bar{\mathbf{v}}, \mathbf{c} \,|\, \mathbf{M})} \quad . \end{aligned} \tag{15}$$

The subscript 'd' in T_d denotes the diagonal d on which block (i,j) locates. In the following calculation of $H_{m,n,l}(i,j)$, the summations are always over state sequences with the same classes as those appeared in the training data.

$$\begin{aligned} & H_{m,n,l}(i,j) \\ = \; & P(s_{i-1,j} = m, s_{i,j-1} = n, s_{i,j} = l \,|\, \bar{\mathbf{v}}, \mathbf{c}, \mathbf{M}) \\ = \; & \begin{cases} \sum_{T_d} \sum_{T_{d-1}} P(T_d, T_{d-1} \,|\, \bar{\mathbf{v}}, \mathbf{c}, \mathbf{M}) & , \text{ if } C(m) = c_{i-1,j}, C(n) = c_{i,j-1}, C(l) = c_{i,j} \\ 0 & , \text{ otherwise} \end{cases} . \end{aligned}$$

We then consider the case $C(m) = c_{i-1,j}$, $C(n) = c_{i,j-1}$, and $C(l) = c_{i,j}$. In the following derivation, the summations over T_d and T_{d-1} are constrained to the T_d satisfying $T_d(i,j) = l$ and the T_{d-1} satisfying $T_{d-1}(i-1,j) = m$, $T_{d-1}(i,j-1) = n$.

$$H_{m,n,l}(i,j) = \sum_{T_d} \sum_{T_{d-1}} \frac{\theta_{T_{d-1}}(\Delta(i,j)-1) \cdot P(T_d \,|\, T_{d-1}, \mathbf{M}) P(\bar{\mathbf{v}}(d) \,|\, T_d, \mathbf{M}) \cdot \beta_{T_d}(\Delta(i,j))}{P(\bar{\mathbf{v}}, \mathbf{c} \,|\, \mathbf{M})} \quad . \tag{16}$$

Although the forward and backward probabilities can significantly reduce the computation for $L_m(i,j)$ and $H_{m,n,l}(i,j)$, computation complexity is still rather high due to the two dimensionality. Equation (13) and (14) for evaluating the forward and backward probabilities are summation over all state sequences on diagonal $d-1$, or $d+1$, with consistent classes with the training data. With

the increase of blocks on a diagonal, the number of state sequences increase exponentially. The same problem happens with the calculation of $L_m(i,j)$ and $H_{m,n,l}(i,j)$. Consequently, we make an approximation in the calculation of $L_m(i,j)$ and $H_{m,n,l}(i,j)$ to avoid computing the backward and forward probabilities. Recall the definitions in section III

$$H^{(p)}_{m,n,l}(i,j) = \sum_{\mathbf{s}} I(m = s_{i-1,j}, n = s_{i,j-1}, l = s_{i,j}) P(\mathbf{s} \,|\, \mathbf{y}, \phi^{(p)}) \quad ,$$

$$L^{(p)}_{m}(i,j) = \sum_{\mathbf{s}} I(m = s_{i,j}) P(\mathbf{s} \,|\, \mathbf{y}, \phi^{(p)}) \quad .$$

To simplify the calculation of $L_m(i,j)$ and $H_{m,n,l}(i,j)$, we will assume that the single most likely state sequence accounts for virtually all the likelihood of the observations. Thus we aim at finding the optimal state sequence which maximizes $P(\mathbf{s} \,|\, \mathbf{y}, \phi^{(p)})$. This is the Viterbi training algorithm. The maximization of $P(\mathbf{s} \,|\, \mathbf{y})$ given model $\phi^{(p)}$ can be achieved by a two dimensional version of the Viterbi algorithm, which is described in next section.

V Variable-state Viterbi Algorithm

Our purpose of using the 2-D Viterbi algorithm is to maximize $P(\mathbf{s} \,|\, \mathbf{y})$, which is equivalent to the maximization of $P(s_{i,j}, \mathbf{v}_{i,j},\ (i,j) \in \mathbb{N})$ constrained to $C(s_{i,j}) = c_{i,j}$ in the training process. When we apply the trained model to classify images (testing process), we also aim at finding states $s_{i,j}$, $(i,j) \in \mathbb{N}$ to maximize $P(s_{i,j}, \mathbf{v}_{i,j},\ (i,j) \in \mathbb{N})$. The states are then mapped into classes. In the testing process, since $c_{i,j}$ is to be decided, the previous constraint is removed.

In the discussion, we consider the unconstrained case, i.e., the testing situation, since in the constrained case, the only difference is to shrink the search range of $s_{i,j}$ to states corresponding to class $c_{i,j}$. We can expand $P(s_{i,j}, \mathbf{v}_{i,j},\ (i,j) \in \mathbb{N})$ as

$$\begin{aligned} P(s_{i,j}, \mathbf{v}_{i,j},\ (i,j) \in \mathbb{N}) &= P(s_{i,j},\ (i,j) \in \mathbb{N}) \cdot P(\mathbf{v}_{i,j},\ (i,j) \in \mathbb{N} \,|\, s_{i,j},\ (i,j) \in \mathbb{N}) \\ &= P(s_{i,j},\ (i,j) \in \mathbb{N}) \cdot \prod_{(i,j) \in \mathbb{N}} P(\mathbf{v}_{i,j} \,|\, s_{i,j}) \\ &= P(T_0) \cdot P(T_1|T_0) \cdot P(T_2|T_1) \cdots P(T_{w+z-2}|T_{w+z-3}) \cdot \prod_{(i,j) \in \mathbb{N}} P(\mathbf{v}_{i,j} \,|\, s_{i,j}) \end{aligned}$$

where T_i denotes the sequence of states for blocks lying on diagonal i. The last equality comes from Equation (4).

Since T_i serves as an "isolating" element in the expansion of $P(s_{i,j}, (i,j) \in \mathbb{N})$, the Viterbi algorithm can be applied straightforwardly to find the combination of states which maximizes the likelihood $P(s_{i,j}, \mathbf{v}_{i,j},\ (i,j) \in \mathbb{N})$. The difference from the normal Viterbi algorithm is that the number of possible sequences of states at every position in the Viterbi transition diagram increases exponentially with the increase of blocks in T_i. If there are M states, the amount of computation and memory are both in the order of M^k, where k is the number of states in T_i. Fig. 3 shows an example. Hence, we refer to this version of the Viterbi algorithm as a variable-state Viterbi algorithm.

The fact that in the two dimension case, only a sequence of states on a diagonal, rather than a single block, can serve as an "isolating" element in the expansion of $P(s_{i,j}, (i,j) \in \mathbb{N})$ causes computation infeasibility for the variable-state Viterbi algorithm. To reduce computation, at every position of the Viterbi transition diagram, the algorithm only uses N out of all the M^k sequences

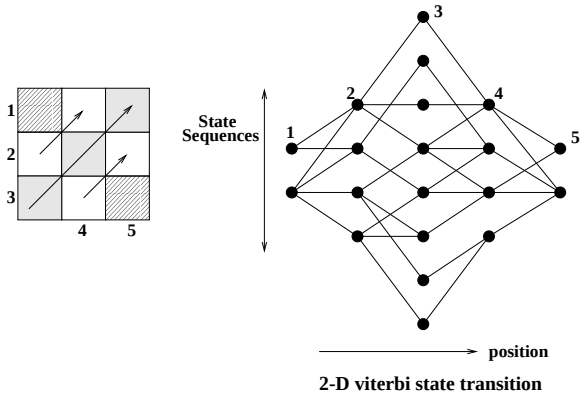

Figure 3: The variable-state Viterbi algorithm

of states, shown in Fig. 4. The paths are constrained to pass one of these N nodes. To choose the N sequences of states, the algorithm separates the blocks in the diagonal from the other blocks by ignoring their statistical dependency. Consequently, the posterior probability of a sequence of states on the diagonal is evaluated as a product of the posterior probability of every block. Then, the N sequences with the largest posterior probabilities are chosen as the N nodes allowed in the Viterbi transition diagram. The implicit assumption in doing this is that the optimal state sequence (the node in the optimal path of the Viterbi transition diagram) yields high likelihood when the blocks are treated independently. We also expect that when the optimal state sequence is not among the N nodes, the chosen suboptimal state sequence coincides with the optimal sequence at most of the blocks. We refer to the sub-optimal version of the algorithm as the path-constrained variable-state Viterbi algorithm. This algorithm is different from the M-algorithm introduced for source coding by Jelinek and Anderson [29] since the N nodes are pre-selected to avoid calculating the posterior probabilities of the M^k state sequences.

A fast algorithm is developed for choosing such N sequences of states. We do not need to calculate the posterior probabilities of all the M^k sequences in order to choose the largest N from them. In the following discussion, we consider the maximization of the joint log likelihood of states with feature vectors, since the maximization of the posterior probability of the states given the feature vectors is equivalent to maximizing the joint log likelihood. Also note that the log likelihood of a sequence of states is equal to the sum of the log likelihood of the individual states because we ignore context information in the pre-selection of nodes. Suppose there are k blocks on a diagonal and each block exists in one of M states. The log likelihood of block i being in state m is $\gamma_{i,m}$. The pre-selection of the N nodes is simply to find N state sequences $\{s_i, i = 1, ..., k\}$ with the largest $\sum_{i=1}^{k} \gamma_{i,s_i}$. Denote $\mathbf{s} = \{s_i, i = 1, ..., k\}$. Suppose we want to find the state sequence $\max_{\mathbf{s}}^{-1} \sum_{i=1}^{k} \gamma_{i,s_i}$, it is unnecessary to calculate $\sum_{i=1}^{k} \gamma_{i,s_i}$ for all the M^k state sequences. We only need to find $\max_{s_i}^{-1} \gamma_{i,s_i}$ for each i, then the optimal state sequence $\mathbf{s} = \{\max_{s_i}^{-1} \gamma_{i,s_i},\ i = 1, ..., k\}$. The idea can be extended for finding the N sequences with the largest log likelihood.

To ensure that the path-constrained variable-state Viterbi algorithm yields results sufficiently close to the variable-state Viterbi algorithm, the parameter N should be larger when there are more blocks in the 2-D Markov chain. As a result, we usually break an image into sub-images to avoid too many blocks in one chain. Every sub-image is assumed to be a 2-D Markov chain, but the dependence between sub-images is ignored. In practice, the degradation of performance is negligible as long as the sub-images are sufficiently global.

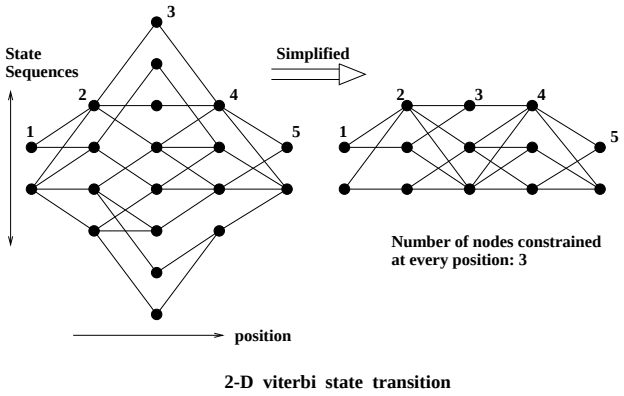

Figure 4: The computation reduced Viterbi algorithm

VI Applications

VI.1 Feature Selection

Choosing features is a critical issue in classification because features often set the limits of classification performance. For a classifier based on the 2-D HMM, we use both intra-block features and inter-block features. The intra-block features are defined basically according to the pixel intensities in a block. They are aimed at describing the statistical properties of the block. Features selected vary greatly for different applications. Widely used examples include moments in spatial domain or frequency domain and coefficients of transformations, e.g., the discrete cosine transform (DCT).

The inter-block features are defined to represent relations between two blocks, e.g., the difference between the average intensities of the two blocks. The use of the inter-block features is similar to that of delta and acceleration coefficients in speech recognition, in which there is ample empirical justification for the inclusion of these features [16]. The motivation for us to use inter-block features is to compensate for the strictness of the 2-D HMM. The 2-D HMM assumes state transition probabilities to be constants. In practice, however, we expect that a transition to a state may depend on some mutual properties of two blocks. For instances, if the two blocks have close intensities, they may be more likely to be in the same state. Since it is too complicated to estimate models with transition probabilities being functions, we retain the constant transition probabilities and offset this assumption somewhat by incorporating the mutual properties into feature vectors such that they can influence the determination of states through posterior probabilities. In the 2-D HMM, since the states of adjacent blocks right above or to the left of a block determine the transition probability to a new state, mutual properties between the current block and these two neighboring blocks are used as inter-block features.

VI.2 Aerial Image Segmentation

The first application of our algorithm is the segmentation of man-made and natural regions of aerial images. The images are 512×512 gray-scale images with 8 bits per pixel. They are the aerial images of the San Francisco Bay area provided by TRW (formerly ESL, Inc.) [30]. An example image and its hand-labeled segmented image are shown in Fig. 6. Four images are used to train a model and another image outside the training set is used for testing. The testing image is the one shown in Fig. 6.

We divide the images into 4×4 blocks and use DCT coefficients or averages over some of them

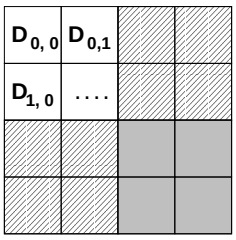

Figure 5: DCT coefficients of a 4×4 block.

as described below as features. There are 6 such features. The reason to use DCT coefficients is that the different energy distributions in the frequency domain can distinguish the two classes more directly. Denote the DCT coefficients for a 4×4 block by $\{D_{i,j}, i, j \in (0, 1, 2, 3)\}$. In Fig. 5, the order of $D_{i,j}$ is given. The definitions of the 6 features are given in the list below.

1. $f_1 = D_{0,0}$; $f_2 = |D_{1,0}|$; $f_3 = |D_{0,1}|$;
2. $f_4 = \frac{\sum_{i=2}^{3} \sum_{j=0}^{1} |D_{i,j}|}{4}$;
3. $f_5 = \frac{\sum_{i=0}^{1} \sum_{j=2}^{3} |D_{i,j}|}{4}$;
4. $f_6 = \frac{\sum_{i=2}^{3} \sum_{j=2}^{3} |D_{i,j}|}{4}$.

We also use the spatial derivatives of the average intensity values of blocks as features. In particular, the spatial derivative refers to the difference between the average intensity of a block and the average intensity of the block's upper neighbor or left neighbor.

Hidden Markov models with different number of states are trained and tested. Simulations show that models with 3 to 6 states for each class yield very similar results. For the result we shall give in this section, we used a model with 5 states for each class. Setting more than 6 states for each class results in worse classification performance for two reasons. One is that the model closest to the truth may not be so sophisticated that each class has more than 6 states. The other reason is that more complicated models require a larger training set. In the case of a fixed training set, the model estimation becomes less accurate with the increase of parameters.

When training and applying the HMM using the path-constrained 2-D Viterbi algorithm, we divide an image into square sub-images each containing 64 blocks. The sub-images are considered as separate Markov chains. The number of nodes constrained at each position in the Viterbi transition diagram, N, is chosen as 32 for the result provided in this section. We experiment on several Ns. For N from 2 to 8, the performance is gradually enhanced. For N greater than 8, the results, with minor differences, start showing a convergence trend. The classification error rate with $N = 8$ is about 1.5% higher than that with $N = 32$. As classification time is spent mainly on the Viterbi searching process, and the Viterbi searching time increases at the order of the second power of the number of nodes at every transition step, the classification time is roughly proportional to N^2. Simulations on one test image show that the CPU time to classify the image on a Pentium Pro 200MHz PC is 17 seconds for $N = 8$, 60 seconds for $N = 16$, and 226 seconds for $N = 32$.

We compare the 2-D HMM result with that obtained by CART [1]. For the CART algorithm, the feature vectors do not include the inter-block features used in the HMM algorithm because the classification is performed independently on blocks. The basic idea of CART is to partition a feature space by a tree structure and assign a class to every cell of the partition. Feature

vectors landing in a cell are classified as the class of the cell. CART is a powerful algorithm for statistical classification. Its applications are not limited to image classification [1]. It enjoys both efficiency and simplicity, which makes the algorithm popular. When we apply the algorithm to image classification, we ignore context information and consider each block independently. We thus use the result as a benchmark for comparison to show that context information can improve classification performance. The classification results for both methods are shown in Fig. 7. Using CART, we obtain error rate of 21.12%. However, the 2-D HMM algorithm achieves error rate of 14.68%. A visual difference to note is that the result of CART appears "noisy", due to the fact that blocks are classified independently and over-localization causes scattered errors.

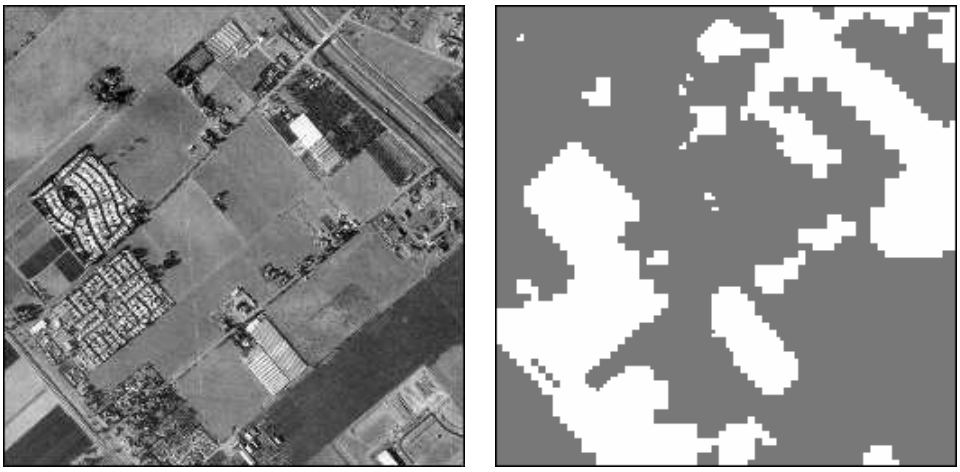

Figure 6: An image and its hand labeled classes. White: man-made, Gray: natural.

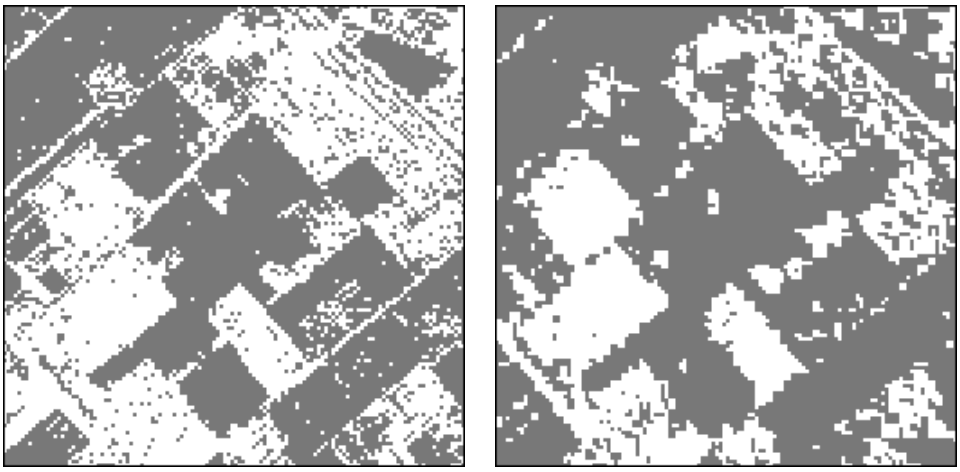

Figure 7: Comparison of the classification results of CART and 2-D HMM. Left: CART with classification error rate **21.12**%, Right: HMM with classification error rate **14.68**%. White: man-made, Gray: natural.

The segmentation of aerial images is also studied by Oehler [30] and Perlmutter [31]. In both cases, The Bayes vector quantizer (BVQ) [30, 31, 32, 33] is used as a classifier. The basic idea of BVQ is to design a vector quantizer to achieve good compression and classification performance simultaneously. Unlike ordinary vector quantizers, which aim at minimizing compression distortion,

the BVQ algorithm defines a new distortion measure as a weighted sum of the compression distortion d and the misclassification penalty B, which is usually the classification error rate P_e.

To design a quantizer minimizing $d+\lambda B$, the conditional probability of being in a particular class given the vector is needed. Since the conditional probabilities are generally unknown in practice, an empirical distribution given by the relative frequencies of the classes in the training sequence is used. In the encoding process, as the empirical distribution obtained from the training data usually does not provide a good estimation for data outside the training set, different approaches are taken to approximate the Bayes risk. A simple solution, which is the approach taken by Oehler, et al. [30, 32, 33], is to replace the Bayes risk by the compression distortion in the encoding process, which is referred to as Bayes VQ with MSE encoding [31], since the mean squared error is normally taken as the compression distortion. The more sophisticated approach taken by Perlmutter [31, 34] generates a posterior estimation of the conditional probabilities of the classes in parallel with the design of the quantizer. This posterior estimation is used by the encoder to evaluate the Bayes risk. Perlmutter [31, 34] showed that the BVQ with posterior estimation achieves much better classification performance than that by the Bayes VQ with MSE encoding. In [31, 34], simulations with different posterior estimations and different weights of classification error in the Bayes risk are performed. It is found that adding a small portion of compression distortion in the Bayes risk often benefits classification results. The same test image as shown in Fig. 6 is used. The best result of the simulations can provide a classification error rate of roughly 21.5%.

VI.3 Document Image Classification

The second application of our algorithm is the classification of document images into the text and the graph class. By pictures, we mean continuous-tone images such as photographs. By text, we mean normal text, tables, and graphs [35]. This type of classification is useful in a printing process for separately rendering different local image types. It is also a tool for efficient extraction of data from image database. The features we use contain the two features described in detail in [35]. The first one is a measure of the goodness of match between the empirical distribution of wavelet coefficients in high frequency bands and the Laplacian distribution. It is defined as a χ^2 statistics normalized by the sample size. The second one measures the likelihood of wavelet coefficients in high frequency bands being composed by highly concentrated values. We also use the spatial derivatives of the average intensity values of blocks as features, which is the same as in the previous application. The block size used is 8×8. The HMM has 4 states for each class. Simulations show that models with 2 to 5 states for each class yield similar results.

We compare the result of HMM with that of CART. The image set is provided by Hewlett Packard, Inc. [36, 31]. They are RGB color images with size around 1600×1300. Each color component is 8 bits per pixel. In our simulation, we only use the illuminance component (i.e., gray-scale images). For most images we tested, both algorithms achieve very low classification error rates, about 2% on average. More differences between the two algorithms appear with one sample image shown in Fig. 8 because the picture region in this image is very smooth at many places, which resembles the text class. The classification results of both CART and the 2-D HMM algorithm for this image are shown in Fig. 9. We can see that the result using HMM is much cleaner than the result using CART, especially in the picture regions. This is expected since the classification based on HMM takes context into consideration. As a result, some smooth blocks in the picture regions, which locally resemble text blocks can be correctly identified as picture.

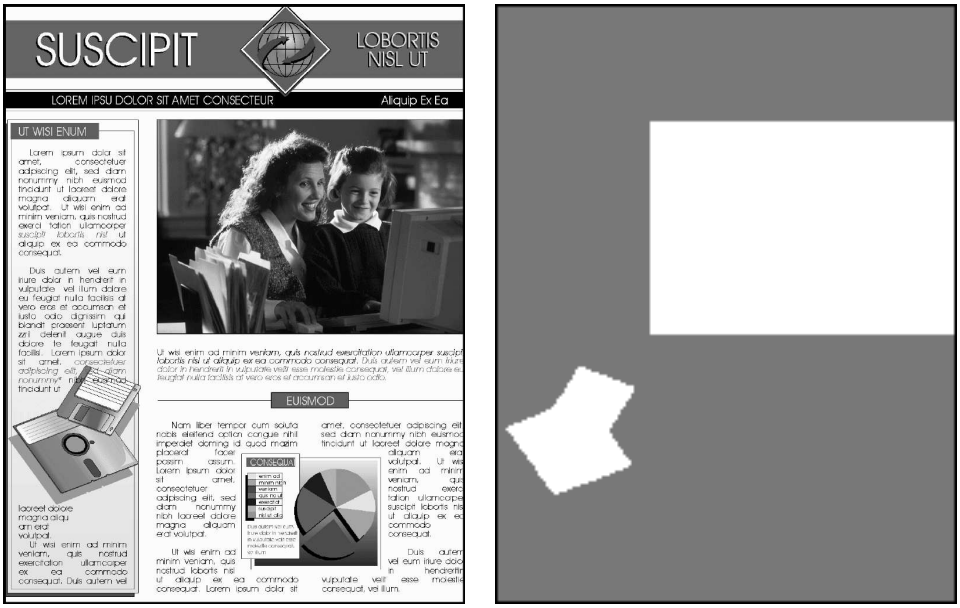

Figure 8: A sample image and its hand-labeled classes. White: photograph, Gray: text.

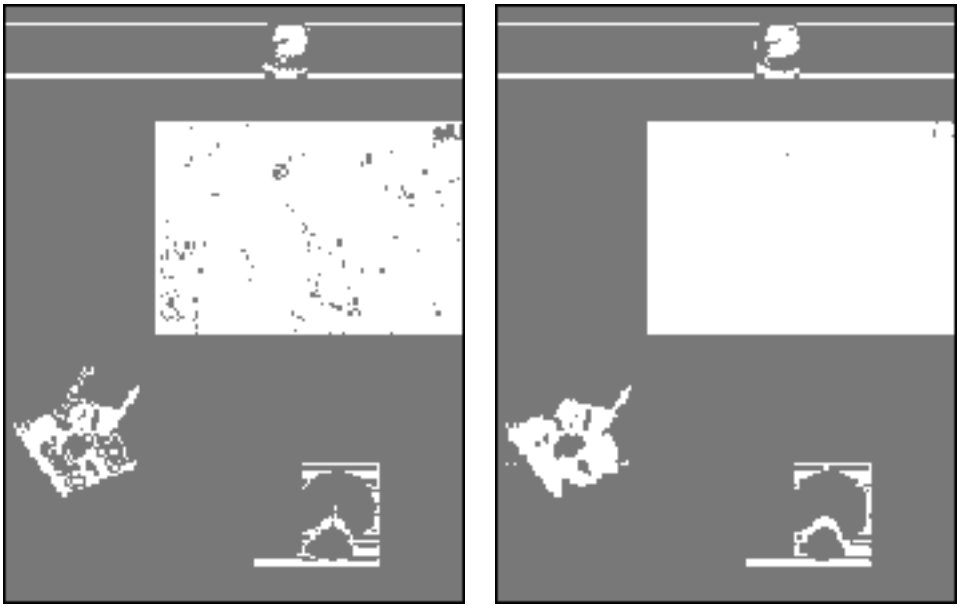

Figure 9: Comparison of the classification results of CART and 2-D HMM. Left: CART classification result. Right: 2-D HMM classification result. White: photograph, Gray: text.

VII Conclusions

We have proposed a two dimensional hidden Markov model for image classification. The two dimensional model provides a structured way to incorporate context information into classification. Using the EM algorithm, we have derived a specific iterative algorithm to estimate the model. As the model is two dimensional, computational complexity is an important issue. Fast algorithms are developed to efficiently estimate the model and to perform classification based on the model. The application of the algorithm to several problems shows better performance than that of existing algorithms.

References

[1] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees,* Chapman & Hall, 1984.

[2] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, pp. 555-585, Kluwer Academic Publishers, 1992.

[3] C. H. Fosgate, H. Krim, W. W. Irving, W. C. Karl, and A. S. Willsky, "Multiscale Segmentation and Anomaly enhancement of SAR Imagery", *IEEE Transactions on Image Processing*, vol. 6, no. 1, Jan. 1997.

[4] J. Li and R. M. Gray, "Context Based Multiscale Classification of Images," *Proceedings of International Conference on Image Processing*, Chicago, Oct. 1998.

[5] L. E. Baum, "An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of Finite State Markov Chains," *Inequalities III*, pp. 1-8, Academic Press, New York, 1972.

[6] L. E. Baum and J. A. Eagon, "An Inequality with Applications to Statistical Estimation for Probabilistic Functions of Markov Processes and to a Model for Ecology," *Bulletin of American Mathematical Statistics*, pp. 360-363, Vol. 37, 1967.

[7] L. E. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *Annals of Mathematical Statistics*, pp. 1554-1563, Vol. 37, 1966.

[8] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Math. Stat.*, pp. 164-171, Vol. 41, No. 1, 1970.

[9] J. K. Baker, "The Dragon System - An Overview," *International Conference on Acoustics, Speech and Signal Processing*, pp. 24-29, Vol. ASSP-23, No. 1, Feb. 1975.

[10] D. B. Paul, "Speech Recognition Using Hidden Markov Models," *The Lincoln Laboratory Journal*, pp. 41-62, Vol. 3, No. 1, 1990.

[11] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[12] X. D. Huang, Y. Ariki and M. A. Jack, *Hidden Markov models for speech recognition*, Edinburgh University Press, 1990.

[13] R. Cole, L. Hirschman, L. Atlas, et al., "The challenge of spoken language systems: research directions for the nineties," *IEEE Transactions on Speech and Audio Processing*, Vol. 3, pp. 1-21, 1063-6676, Jan. 1995.

[14] A. A. Markov, "An Example of Statistical Investigation in the Text of 'Eugene Onyegin' Illustrating Coupling of 'Tests' in Chains," *Proc. Acad. Sci. St.*, Petersburg, VI Series 7, pp. 153, 1913.

[15] R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley & Sons, Inc., 1968.

[16] S. Young, J. Jansen, J. Odell, D. Ollason and P. Woodland, *HTK - Hidden Markov Model Toolkit*, Cambridge University, 1995.

[17] A. J. Viterbi and J. K. Omura, "Trellis Encoding of Memoryless Discrete-time Sources with a Fidelity Criterion," *IEEE Trans. Inform. Theory*, IT-20, pp. 325-332, May 1974.

[18] A. P. Dempster, N. M. Laird and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Roy. Statist. Soc.*, pp. 1-21, Vol. 39, No. 1, 1977.

[19] S. S. Kuo and O. E. Agazzi,"Machine Vision for Keywod Spotting Using pseudo 2D Hidden Markov Models," pp. 81-84, Vol. 5, *ICASSP* 1993.

[20] C. C. Yen and S. S. Kuo, "Degraded documents recognition using pseudo 2d hidden Markov models in gray-scale images," *Proceedings of the SPIE*, Vol. 2277, pp. 180-91, 1994.

[21] R. Kindermann, and J. L. Snell, *Markov Random Fields and Their Applications*, American Mathematical Society, 1980.

[22] S. Geman, and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721-41. Nov. 1984.

[23] K. Abend, T. J. Harley, and L. N. Kanal, "Classification of Binary Random Patterns," *IEEE Transactions on Information Theory*, vol. IT-11, no. 4, pp. 538-544, Oct. 1965.

[24] L. N. Kanal, "Markov mesh models", *Image Modeling*, New York: Academic, 1980.

[25] J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems (with Discussion)," *Journal Royal Statistics Society*, series B, vol. 34, pp. 75-83, 1972.

[26] D. K. Pickard, "A Curious Binary Lattice Process," *J. Appl. Prob.*, vol. 14, pp. 717-731, 1977.

[27] C. F. J. Wu, "On the Convergence Properties of the EM Algorithm," *The Annals. of Statistics*, pp. 95-103, Vol. 11, No. 1, 1983.

[28] P. J. Bickel and K. A. Doksum, *Mathematical Statistics: Basic Ideas and Selected Topics*, Prentice Hall, Englewood Cliffs, NJ, 1977.

[29] F. Jelinek and J. B. Anderson, "Instrumentable Tree Encoding of Information Sources," *IEEE Trans. Inform. Theory*, IT-17, pp. 118-119, Jan 1971.

[30] K. L. Oehler, "Image Compression and Classification Using Vector Quantization," *Ph.D thesis*, Stanford University, 1993.

[31] K. O. Perlmutter, "Compression and Classification of Images Using Vector Quantization and Decision Trees," *Ph.D thesis*, Stanford University, 1995.

[32] K. L. Oehler and R. M. Gray, "Combining Image Classification and Image Compression Using Vector Quantization," *Proceedings of the 1993 IEEE Data Compression Conference*, pp. 2-11, Snowbird, Utah, March 1993.

[33] K. L. Oehler and R. M. Gray, "Combining Image Compression and Classification Using Vector Quantization," *IEEE Trans. PAMI*, 17(5), pp. 461-473, May 1995.

[34] K. O. Perlmutter, S. M. Perlmutter, R. M. Gray, R. A. Olshen, and K. L. Oehler, "Bayes Risk Weighted Vector Quantization with Posterior Estimation for Image Compression and Classification," *IEEE Transactions on Image Processing*, vol. 5, no. 2, pp. 347-60, February 1996.

[35] J. Li and R. M. Gray, "Text and Picture Segmentation by the Distribution Analysis of Wavelet Coefficients," *Proceedings of International Conference on Image Processing*, Chicago, Oct. 1998.

[36] K. O. Perlmutter, N. Chaddha, J. B. Buckheit, R. M. Gray, and R. A. Olshen, "Text Segmentation in Mixed-mode Images Using Classification Trees and Transform Tree-structured Vector Quantization," *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 2231-2234, Vol. 4, Atlanta, GA, May 1996.